View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Sunderland University Institutional Repository

Jaf, Sardar, Alabbas, Maytham and Khudeyer, Raidah S. (2018) Combining Machine Learning Classifiers for the Task of Arabic Characters Recognition. In: International Journal of Asian Language Processing, 2017, Malaysia.

Downloaded from: http://sure.sunderland.ac.uk/id/eprint/10450/

Usage guidelines

Please refer to the usage guidelines at http://sure.sunderland.ac.uk/policies.html or alternatively contact sure@sunderland.ac.uk.

ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/327844402

Combining Machine Learning Classifiers for the Task of Arabic Characters Recognition

Article · September 2018

CITATIONS
0

READS
2,024

3 authors:

Sardar Jaf
Durham University
19 PUBLICATIONS **20** CITATIONS
SEE PROFILE

Maytham Alabbas
University of Basrah
25 PUBLICATIONS **62** CITATIONS
SEE PROFILE

Raidah Khudeyer
University of Basrah
3 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Native language identification from English text View project

Infill Drilling Optimization View project

All content following this page was uploaded by Maytham Alabbas on 24 September 2018.

The user has requested enhancement of the downloaded file.

Combining Machine Learning Classifiers for the Task of Arabic Characters Recognition

Sardar Jaf†, Maytham Alabbas*, and Raidah S. Khudeyer*
†Department of Computer Science, University of Durham, Durham, United Kingdom
*Department of Computer Science, University of Basrah, Basrah, Iraq
sardar.jaf@durham.ac.uk
ma@uobasrah.edu.iq
raidah.khudayer@uobasrah.edu.iq

Abstract

There is a number of machine learning algorithms for recognizing Arabic characters. In this paper, we investigate a range of strategies for multiple machine learning algorithms for the task of Arabic characters recognition, where we are faced with imperfect and dimensionally variable input characters. We show two different strategies to combining multiple machine learning algorithms: manual backoff strategry and ensemble learning strategy. We show the performance of using individual algorithms and combined algorithms on recognizing Arabic characters. Experimental results show that combined confidence-based strategies can produce more accurate results than each algorithm produces by itself and even the ones exhibited by the majority voting combination.

Keywords

Characters recognition, Arabic characters recognition, Optical characters recognition (OCR), kNN, SVM, PNN, ensemble learning, hand written characters recognition.

1. Introduction

Systems combination is a popular approach to improving accuracy in different tasks. This approach involves combining multiple systems,to perform a given task by by exploiting the unique advantage of individual systems to reduce some of the random errors produced by some of them. Systems combination has been widely applied in different fields such as natural language processing (NLP) (Alabbas and Ramsay, 2014) and pattern recognition (Giacinto and Fumera, 2000). In this paper, we evaluate different strategies for combining machine learning classifiers for Arabic characters recognition. Arabic characters (and

modified versions of Arabic characters) are used in various languages sucha as Arabic, Persian, Urdu, Kurdish and others; more than half a billion people use the Arabic characters for writing.

Optical character recognition (OCR) is one of the most successful applications of automatic pattern recognition. OCR is the process of translating a graphical document into a format (i.e., textual document) that a computer can process. This process is used in a number of sub-disciplines of Computer Science, such as image processing; pattern recognition; natural language processing; and artificial intelligence. A comprehensive review of OCR, its histories and development is given in Amin (1998).

The main objective in character recognition task is to accurately recognize handwritten or typist characters to facilitate human-machine interaction. The motivation behind developing character recognition systems is inspired by their wide range of applications including archiving documents, automatic reading of checks, and number plate reading. Despite the large quantity of existing research in this field there is no obvious mathematical function that can perform this translation. Considerable attention has been paid to Latin and Chinese characters recognition, while Arabic characters recognition still received limited attention in spite of the challenge due to the difficulties of these characters, which stems from the characteristics of the Arabic characters and the way these characters are connected and written (Supriana and Nasution, 2015).

In this paper, several combining strategies for the recognition of isolated printed Arabic characters are investigated. These strategies are based on combining three machine learning classifiers (k-Nearest Neighbors (kNN), Support Vector Machine (SVM), and Probabilistic Neural Network (PNN)) as classifiers and then using different decision-making strategies such as majority vote, confidence systems, and others to predict the correct class for a character. Simulation results prove that the combination strategies often produce higher recognition rate compared with the using individual classifiers. The remainder of this paper is organized as follows: in Section 2 the Arabic characters are analyzed and some of the main problems related to recognizing these characters are presented in addition to describing some previous works on Arabic characters recognition. A brief explanation of the three recognition techniques is given in Section 3. Section 4 explains the combination strategies and the experiments performed using these strategies. The results of the experiments obtained from different strategies are presented in Section 5. Finally, Section 6presents the conclusions and future works.

2. Features of the Arabic Characters

Arabic writing and Arabic characters have many features that make an Arabic characters

recognition system different from character recognition systems for other languages such as Latin and Chinese. The Arabic language is written from right to left in a cursive script in both handwritten and typewritten. Arabic characters also have many characteristics that complicate the recognition of such characters. Some of the key characteristics are listed below:

1. The Arabic alphabet consists of 29 characters. Each Arabic character might have up to four forms depending on its relative position in the word (i.e., begin, middle, end, and isolated). The relative position of a character in a word increases the number of patterns from 29 to about 100 patterns. Table 1 shows the Arabic character patterns (each table cell contains one character with its possible forms).
2. The majority of the Arabic characters (around 16 out of 29 characters) have a character complementary that is associated with the body of the character. This complementary may be a dot, two dots, three dots, or zigzag (hamza). It can be above the character (such as ف), below (such as ب), or inside the character (such as ج).
3. There are different groups of characters that have the same body, but they are distinguished by the number of complementary dots (such as ث ,ت,ن), the position of dots whether they are above or below the characters (as in ,خ،), the present of dot(s) (such as ذ, د), or the shape of a character complementary whether it is a dot or zigzag (as in ـئـ ، ـنـ).
4. Both widths and heights of Arabic characters are variable (e.g. ا and ب).

Meem	Ayn	Seen	HHa	Hamza
مـمـم م	عـعـع ع	سـسـس س	حـحـح ح	ء ئ ئـ
Noon	Ghayn	Sheen	Khaa	Alif
نـنـن ن	غـغـغ غ	شـشـش ش	خـخـخ خ	ا ـا
Ha	Faa	Saad	Daal	Baa
هـهـه ه	فـفـف ف	صـصـص ص	د ـد	بـبـب ب
Waaw	Qaaf	Dhad	Dhaa	Taa
و ـو	قـقـق ق	ضـضـض ض	ذ ـذ	تـتـت ت
Yaa	Kaaf	Taa	Raa	Thaa
يـيـي ي	كـكـك ك	طـط	ر ـر	ثـثـث ث
	Laam	Dhaa	Zaay	Jeem
	لـلـل ل	ظـظ	ز ـز	جـجـج ج

Table 1: Arabic character patterns

In 1999, Bazzi, Schwartz, and Makhoul (1999) propsed a complete Arabic OCR system based on Hidden Markov Model (HMM) classifier. They reported a character error rate of just over 3%. This high performance is mainly because of the strong preprocessing

and post-processing modules in their system. Khorsheed (2007) proposed an Arabic multi-font OCR system using discrete HMMs along with intensity based features. He implemented character models using mono and tri models. The experiments were mainly on simplified Arabic font and a character accuracy of just under 78% was reported. In a similar way, Attia, Rashwan, and El-Mahallawy (2009) proposed a system to recognise Arabic characters by using descrete HMMs with new features. Because they used a synthesized and very clean dataset as well as high quality images (which were scanned at 600dpi), they achieve a high accuracy rate of 99.3%. Furthermore, in 2012, Rashwan et al (2012), proposed a an HMM-based system using bi-gram and 4-gram character language model. They evaluated their system on a ALTIC dataset[1] and reported 84% and 88% for bi-gram and 4-gram character language models respectively.

3. Recognition Methodologies

In this study, several machine learning classifiers were chosen to classify printed Arabic characters, shown in Table 1. In the following subsections, a functional description of these techniques is introduced (Martinez and Martinez, 2015).

a) k-Nearest Neighbors

K-Nearest Neighbor (kNN) is a simple machine learning algorithm, which is used for classifying objects based on the nearest training sets in the feature space. An object is classified by a majority vote of its neighbors, with the object being assigned to a class most common amongst its k nearest neighbor, where k is a small positive integer.

b) Support Vector Machine

The Support Vector Machine (SVM) classifier is based on statistical learning theory. It is a powerful and versatile machine learning algorithm, which can particularly be useful for classification of small-, or medium-size, datasets.. The standard SVM takes a set of input data and predicts, for each given input, the possible classes form the input. The process of rearranging the objects is known as mapping. After learning by quadratic programming (QP), the samples of non-zero weights are called support vectors (SVs).

c) Probabilistic Neural Network

Probabilistic Neural Network (PNN) is an implementation of a statistical algorithm called kernel discriminant analysis. The PNN architecture is composed of many interconnected

1 Available at: http://www.altec-center.org/conference/?page_id=84

processing units, or neurons, organized in four successive layers: input layer, pattern layer, summation layer, and output layer. The input layer does not perform any computation; it simply distributes the input to the neurons in the pattern layer. The pattern layer contains one neuron for each case in the training data set. It stores the values of the predictor variables for the case along with the target value. The summation layer performs an average operation of the outputs from the pattern layer for each class. The output layer performs a weighted vote, selecting the largest value and uses the largest vote to predict the target category.

d) Ensemble Learning

Ensemble learning is a machine learning technique where multiple machine learning algorithms (learners) are trained to solve a particular problem. The models that are generated by training different learners on a dataset are combined to perform as a single unit. Specifically, a system is normally constructed by training different learners in parallel or in sequential styles where the output of one learner has influence on subsequent learners. Then, the base learners are combined to use one of several combination schemes: in this study, we experiment with the following two schemes:

- Voting classifiers. The main idea behind this scheme is to train different (possibly weak) classifiers and use them to make predictions. The prediction of the different classifiers are aggregated using different voting methods: hard voting and soft voting. In hard voting method, the predictions produced by all the classifiers are aggregated and the prediction with the most votes is selected as the final prediction. In soft voting method, the argmax of the sum of predicted probabilities is used for predicting the class labels. Using the soft voting method learner must support probability prediction.
- Boosting. The general idea behind this ensemble scheme is to sequentially train different learning algorithms and each algorithm corrects the decision made by its predecessor.

4. Experiment Results

As we mentioned before, the aim of this research is to investigate and evaluate a range of strategies for combining different machine learning classifiers for an Arabic character recognition system, where the input characters are imperfect and dimensionally variable, and compare the results of combining different classifiers with individual classifiers.

Each recognition system, here, is conducted through three main modules. The first module is responsible for preparing the input images by acquisition and digitizing of the

image, remove noise, binarize, and thinning. The second module extracts the main features of the preprocessed images. The third module processes the main features to recognize the input characters. Several systems are used in this module; individual systems such as kNN, SVM, PNN and multiple combined systems. A brief description of each module is in the following lines.

a) Preprocessing

The preprocessing step involves eliminating some variability related to the writing process, such as the variability due to the writing environment, writing style, acquisition and digitizing of the image. The main steps of the preprocessing module are as below:

1) Noise reduction

Images usually contain noise. One approach to reduce noise is to apply adaptive median filter (Zhao and Li, 2007). The advantage of the median filter is it keeps the edges of the image but it eliminates some of the noise.

2) Binarizing

This part is responsible for converting the input image to binary image. This is done by replacing all pixels in the input image with luminance greater than a specific level with the value 1 for white color and the value 0 for black color.

3) Thinning

Thinning is done to make the characters around one pixel wide.

b) Feature Extraction

The extracted characters from the input image have different dimensions (e.g., the width of the Arabic character ب is different from the width of the Arabic character ا, and the same for the height). To deal with this challenge, the discrete cosine transform (DCT) is adopted to extract the features of the characters. DCT is a technique used for converting the image data to its elementary frequency components where high-value coefficients are clustered in the upper left corner and low-value coefficients in the bottom right of the resulted matrix. In order to improve the performance and efficiency of the recognition system, we have investigated three various feature extractors, e.g., 10 coefficients, 20 coefficients, and 64 coefficients. Indeed, each extractor type range is different from those of the other extractor types. Therefore, each feature extractor extracts vector which is not uniform with the other vectors extracted from other feature extractors. We do not have enough space to include all the details and the results of these feature extractors. Instead, for simplicity, we focus on the

64 DCT coefficients feature extractor, which has been the most useful one in practice. We apply the DCT on a character and select the first 64 higher value DCT coefficients, which are extracted in a zigzag fashion as a feature vector for recognizing this character.

c) Recognition technique

We have developed different systems for the task of Arabic characters recognition. Below is a list of 15 different systems that have been investigated:

- **System$_1$**: this system uses the kNN classifier in isolation with the number of nearest neighbors (k=1). The output is an integer number that represents a character, e.g., 1=all patterns of Alif,…,28=all patterns of Yaa.
- **System$_2$**: this system uses the SVM classifier in isolation, which relies on multi-class SVM (28 SVMs, one-rest method) with the order of polynomial kernel equal to 2.
- **System$_3$**: this system uses the PNN in isolation with spread of radial basis functions equal to 0.2. The output is an integer number that represents a character, e.g., 1=all patterns of Alif,…,28=all patterns of Yaa.
- **System$_4$**: this system accepts the result of System$_1$ and System$_2$ if they agree and backoff to System$_3$ if they do not (whether or not the backoff system agrees with either of the chosen pair).
- **System$_5$**: this system accepts the result of System$_1$ and System$_3$ if they agree and backoff to System$_2$ if they do not (whether or not the backoff system agrees with either of the chosen pair).
- **System$_6$**: this system accepts the result of System$_2$ and System$_3$ if they agree and backoff to System$_1$ if they do not (whether or not the backoff system agrees with either of the chosen pair).
- **System$_7$**: this system accepts the result of System$_1$ and System$_2$ if they agree and backoff to the most confident system (1 or 2) if they do not agree.
- **System$_8$**: this system accepts the result of System$_1$ and System$_3$ if they agree and backoff to the most confident system (1 or 3) if they do not agree.
- **System$_9$**: this system accepts the result of System$_2$ and System$_3$ if they agree and backoff to the most confident system (2 or 3) if they do not agree.
- **System$_{10}$**: this system accepts the result of at least two systems if they agree and backoff to the most confident system (1-3) if they do not agree.
- **System$_{11}$**: this system accepts the result of System$_1$, System$_2$, and System$_3$ if they agree and backoff to the most confident system (1-3) if they do not agree.
- **System$_{12}$**: this system accepts the result of the most confident system only.

- **System$_{13}$**: This system is based on ensemble learning using the hard voting method.
- **System$_{14}$**: This system is based on ensemble learning using the soft voting method.
- **System$_{15}$**: This system is based on ensemble learning using Gradient-boosting classifier.

The learning algorithms in System1 to System 12 are based on kNN, SVM, and PNN. The learning algorithms in System$_{13}$ and System$_{14}$ consist of Random Forest, Knn and SVM. Systems$_{15}$ uses 100 decision stump as weak learners.

In the systems (7-12), we used a technique that depends on using the system which is known to be the most reliable system for each Arabic character. After testing the individual systems on the test set with different levels of noise, we find the most reliable system for each Arabic character and then used these confidence levels to decide how much each system should be trusted for each character. We find that, for instance, System$_1$ should be trusted when the character is 'ر', whereas System$_2$ should be trusted when the character is 'ا'. Consider a situation where three systems completely disagree to recognize a character. For example, System$_1$ decides the character is 'ر' with confidence level for classifying this character equal to 90%, System$_2$ decides the character is 'د' with confidence level for classifying this character equal to 80%, and System$_3$ decides the character is 'ذ' with confidence level for classifying this character equal to 75%. In this case, the final decision will be the character 'ر' because System$_1$ has the highest confidence level for classifying such character.

5. Results

We performed experiments using the individual systems (1-3) above by training them on the training set, which is all the Arabic characters in Table 1 (except Hamza). In order to make our experiments more realistic, all systems were tested on the test set, which is the same training characters set but have been corrupted by three levels of "Salt & Pepper" noise (i.e. 10%, 30%, and 50%). We have allocated 80% of the data for training and 20% for testing.

The results of these experiments, in terms of recognition rate, are illustrated in Table 2. As can be seen in the table below, System$_1$ achieves the best result.

System	Recognition rate for the noise level		
	10%	*30%*	*50%*
System$_1$	**96%**	**92%**	64%
System$_2$	90%	82%	**69%**
System$_3$	94%	86%	48%

Table 2: Isolated systems recognition rates.

If one has multiple classifier systems and they all suggest a specific character, intuitively, the best action is to accept that suggestion. The key issue is which action to take (i.e., what character to accept) if all the systems disagree with each other. Before investigating this issue, it is worth looking at what happens when they do agree.

We, therefore, measure the systems' performance using precision (P), recall (R) and F-score (F) as a metric measure for various combinations of systems on cases where they agreed.

Table 3 shows the precision, recall, and F-score for the merge of the systems output where they agree, either pairwise or unanimously.

Systems	Noise Level (10%)			Noise Level (30%)			Noise Level (50%)		
	P%	*R%*	*F*	*P%*	*R%*	*F*	*P%*	*R%*	*F*
$System_1$+ $System_2$	**100**	86	0.93	97.4	76	0.85	85	51	0.64
$System_1$+ $System_3$	96.9	93	0.95	94.4	85	0.90	71.8	51	0.60
$System_2$+ $System_3$	98.8	85	0.91	97.3	73	0.83	65.5	36	0.47
Three systems agree	**100**	84	0.93	**98.6**	72	0.83	97.2	35	0.52
Two or more system agree	97	**96**	**0.97**	94.7	**90**	**0.92**	73.5	61	0.67
$System_{13}$	96	82	0.88	97.7	71	0.81	72	38	0.43
$System_{14}$	99	89	0.94	97.3	75	0.84	**98**	**64**	**0.73**
$System_{15}$	97	62	0.72	88.6	59.3	0.66	97	42	0.50

Table 3: Precision (P), recall (R) and F-score (F)
for agreement output of two or more systems

Not surprisingly, the precision on combining systems is considerably higher than the precision of any individual system. More importantly, when we combine only two systems, we find that the combination of $System_2$ with either of $System_1$ or $System_3$ gives better precision and lower recall than combining $System_1$ and $System_2$. This is slightly surprising: $System_2$ uses a different technique from the systems (1 and 3), and hence when it agrees with one of them it is likely that they have arrived at the same conclusion by different

routes, and hence this conclusion has good supporting evidence.

A system, however, is required to give a complete recognition result, so we have to recommend a backoff strategy for cases where not all of the systems agree. Here we briefly consider two promising strategies for dealing with this challenge-taking the output if all three systems agree (highest precision in Table 3) and taking the output if any pair agree (highest F-score in Table 3)–and investigate a range of backoff strategies.

The backoff strategies are divided to two groups: (i) voting-based backoff group which is a set of voting strategies; and (ii) confidence-based backoff group which is a set of techniques based on identifying which system is best at dealing with particular kinds of characters. The latter group has proved highly effective for combining POS taggers (Alabbas and Ramsay, 2012), and parsers (Alabbas and Ramsay, 2011), and it seemed *prima facie* plausible that it would also work for character recognition.

Table 4 shows the results obtained from combining multiple systems for different noise levels of the test set. As noted above, we find that the combined systems (7-15) outperform each of the individual systems (1-3, as in Table 2), and they also achieve better recognition rates than the simple voting-based backoff systems (4-6).

System	Recognition rate for the noise level		
	10%	*30%*	*50%*
$System_4$	96%	90%	61%
$System_5$	96%	90%	61%
$System_6$	96%	90%	61%
$System_7$	**100%**	**98%**	81%
$System_8$	97%	93%	64%
$System_9$	99%	95%	78%
$System_{10}$	96%	90%	61%
$System_{11}$	**100%**	**98%**	81%
$System_{12}$	**100%**	**98%**	81%
$System_{13}$	97%	95%	89%
$System_{14}$	98%	96%	**93%**
$System_{15}$	92%	88%	87%

Table 4: Recognition rates for combining systems, deferent noise levels.

6. Conclusions and Future Work

Intuitively, having multiple systems performing the same task. we can assume that better performance can be obtained by combining the output of different systems than using the

output of individual system. Strategies for multi-system collaboration have been proposed, with majority voting being particularly popular. We have investigated here a range of strategies for combining machine learning classifiers for Arabic characters recognition: the best strategy we have found for recognizing the Arabic characters involves examining the confidence level of each system, and accepting the output given by the most confident system. We hypothesize that the reason for the effectiveness of this strategy for character recognition arises from the fact that individual systems work in essentially different ways (e.g., different underlying algorithms), and hence if they make systematic errors these will tend to be different. This means, in turn, that the places where they do not make mistakes will be different.

Based on the encouraging findings in this work, we have identified two further research tasks. First, in order to improve the recognition efficiency and generality of the presented systems, we intend to evaluate these systems on multi-font and multi-size training and test sets. Second, we will extend the current systems to deals with Arabic text rather than isolated characters by adding segmentation module to split an input text into words and then into characters.

8. References

Attia, M., Rashwan, M., A., A., and El-Mahallawy, M., A., M., 2009, Autonomously Normalized Horizontal Differentials as Features for HMM-Based Omni Font-written OCR Systems for Cursively Scripted Languages. In *2009 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, pp. 185–190.

Alabbas, M., and Ramday, A., 2014, Combining strategies for tagging and parsing Arabic, In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP 2014)*, pp. 73-77, Doha, Qatar.

Alabbas, M., and Ramsay, A., 2012, Improved POS-Tagging for Arabic by Combining Diverse Taggers, In *Artificial Intelligence Applications and Innovations: 8th IFIP WG 12.5 International Conference, AIAI 2012, Halkidiki, Greece, September 27-30, 2012, Proceedings, Part I*, L. Iliadis, I. Maglogiannis, and H. Papadopoulos, Eds., ed Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 107-116.

Alabbas, M., and Ramsay, A., 2014, Improved Parsing for Arabic by Combining Diverse Dependency Parsers, In *Human Language Technology Challenges for Computer Science and Linguistics: 5th Language and Technology Conference, LTC 2011, Poznań, Poland, Revised Selected Papers, Z. Vetulani and J. Mariani, Eds., ed Cham: Springer International Publishing*, pp. 43-54.

Bazzi, I., Schwartz., R., and Makhoul, J., 1999, An Omnifont Open-Vocabulary OCR System for

English and Arabic. IEEE Transaction on Pattern Analysis and Machine Intelligence 21(6), pp. 495-504.

Giacinto, G., F., G., Roli, F., G., and Fumera, G., 2000, Selection of Classifiers Based on Multiple Classifier Behaviour, In *Advances in Pattern Recognition: Joint IAPR International Workshops SSPR 2000 and SPR 2000 Alicante, Spain, August 30 – September 1, 2000 Proceedings*, F. J. Ferri, J. M. Iñesta, A. Amin, and P. Pudil, Eds., ed Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 87-93.

Khorsheed, M., S. 2007., Offline Recognition of Omnifont Arabic Text Using the HMM Toolkit (HTK). Pattern Recognition Letters 28(12), pp. 1563–1571

Supriana I., and Nasution, A., 2013, Arabic Character Recognition System Development, In *Procedia Technology,* vol. 11, pp. 334-341.

Martinez W., and Martinez, A., 2015, *Computational Statistics Handbook with MATLAB*, 3rd ed.: Chapman and Hall/CRC.

Rashwan, A., M., Rashwan, M., A., Ahmed., A., Abdou,, S., Khalil, A.H., 2012, A Robust Omnifont Open-Vocabulary Arabic OCR System Using Pseudo-2D-HMM. In Proceedings of SPIE – The International Society for Optical Engineering. 8297.

Zhao, Y., and Li, D., and Li, Z., 20017, Performance enhancement and analysis of an adaptive median filter, In *International Conference on Communications and Networking (CHINACOM '07)*, pp. 651-653 China, Shanghai.

View publication stats